Peer-to-Peer Netw. Appl.
DOI 10.1007/s12083-010-0074-2

Sorcery: Overcoming deceptive votes in P2P content sharing systems

Ennan Zhai · Huiping Sun · Sihan Qing · Zhong Chen

Received: 28 December 2009 / Accepted: 3 June 2010

Abstract Deceptive voting behaviors of malicious users are known as the main reason of causing content pollution in Peer-to-Peer (P2P) content sharing systems. Due to the nature of P2P overlay network such as self-organization and anonymity, the existing methods on identifying deceptive votes are not effective, especially for collusive attackers. This paper presents Sorcery, a novel active challenge-response mechanism based on the notion that one side of interaction with the dominant information can detect whether the other side is telling a lie. To make each client obtain the dominant information, our approach introduces the social network to the P2P content sharing system; therefore, clients can establish the friend-relationships with the users who are either acquaintances in reality or those reliable online friends. Using the confidential voting histories of friends as own dominant information, the client challenges target content providers with the overlapping votes of both his friends and the target content provider, thus detecting whether the content provider is a deceptive user. Moreover, Sorcery provides the punishment mechanism which can reduce the impact brought by deceptive voting behaviors, and our work also discusses some key practical issues. The experimental results illustrate that Sorcery can effectively overcome the problem of deceptive voting behaviors in P2P content sharing systems, and work better than the existing reputation models.

Keywords Peer-to-Peer · Social network · Deceptive votes · Content pollution · Challenge-response

E. Zhai · H. Sun · S. Qing · Z. Chen
School of Software and Microelectronics, Peking University, Beijing, China

E. Zhai · H. Sun · S. Qing · Z. Chen
Key Laboratory of Network and Software Security Assurance, Beijing, China

E. Zhai
e-mail: zhaien@infosec.pku.edu.cn

S. Qing
e-mail: qsihan@ss.pku.edu.cn

Z. Chen
e-mail: chen@ss.pku.edu.cn

H. Sun (✉)
Peking University, Room 1530, Science Building 1#, Beijing, 100871, China
e-mail: sunhp@ss.pku.edu.cn

1 Introduction

Peer-to-Peer (P2P) content sharing systems, e.g., BitTorrent [5], KaZaA [3], eMule [4], etc., have become increasingly popular. However, due to the nature of P2P overlay networks, such as anonymity and self-organization, the participants have to face some potential risks involved in the application transactions without adequate experience and knowledge of other users. Many studies indicated that P2P content sharing systems are highly vulnerable to *deceptive voting behaviors* [13, 21].

Deceptive voting behaviors In a typical deceptive voting behavior, individual or collusive malicious users first publish lots of content that may contain invalid meta-data or corrupt data; meanwhile they also download some content from system. Then, they cast

Published online: 22 June 2010

incorrect votes on the content in system to mislead other users. For example, they cast "positive vote" on some corrupt content to confuse normal users. Unable to distinguish authentic content from the corrupt one, the normal P2P users download the undesirable content into their folders. Therefore, we say these normal users are deceived, and we call these malicious users who cast incorrect votes on the content in P2P content sharing system as *deceivers*. As a result, the corrupt content spreads through the P2P network with an extraordinary speed, and causes the availability of P2P content sharing system to be low.

Generally, some previous studies on identifying deceptive voting behaviors mainly focused on the reputation models. However, due to the nature of reputation models, such as *passive aggregation of experiences*, the client is easy to become a victim when encountering the collusive deceivers or individual tricky deceiver.[1] The above situation can be explained based on the principle of Game theory [17, 18] that the adversaries in system sit on the *dominant position*, and the solution is that we need to achieve the conversion of the dominant position through constructing our own *dominant information*. The fundamental insight driving our work is that social network can help the users construct the confidential friend-relationships, and we may treat the confidential information (e.g., content, voting history, etc.) of friends as the dominant information because the friend information is owned by the client only. Therefore, the client can detect whether the content provider is a deceiver using the overlapping voting histories of the content of both his friends and the content provider.

Based on the above analysis, this paper introduces Sorcery, a novel challenge-response mechanism using social network to construct the dominant information for clients. The *challenge* denotes the query about the votes of some content, and the *response* denotes the response messages to "answer" the challenge (The details are mentioned in Section 3.2). Sorcery encompasses *three* key techniques to detect and punish the deceivers in P2P content sharing systems:

Social network Sorcery introduces social network into the P2P content sharing system; therefore, each client can establish his own friend-relationships. These friends share their own information (e.g., content, votes, etc.) with the client, and the friend information of the client is confidential to other peers in the system. Specifically, *Alice*'s friend information cannot be seen by any user (except *Alice*) in the system.

Challenge-response mechanism Sorcery clients utilize the overlapping voting histories of both his friends and the content provider to challenge the latter actively, and judge whether the other side is a deceiver or not based on the correctness of his response. Besides, the client assigns each user in the system a reliability degree. According to the result of each challenge-response, the client will adjust the reliability degree of the other side.

Punishment mechanism Sorcery client ranks each search result based on the honesty (the reliability degree) of the *target content provider*; therefore, the probability of impact brought by deceivers is reduced. In other words, because deceivers' reliability degrees normally should be lower than normal users', the content provided by deceivers will certainly be placed at the end of client's search result.

To make Sorcery easier to be adopted into the real-world system, we discuss the solutions for three issues which possibly happen in practical application (The details are mentioned in Section 3.4). In addition, we also discuss the solutions of Sorcery on fighting against the Man-In-The-Middle (MITM), Sybil [22], Denial-of-Service (DoS) and the content integrity attacks.

To evaluate the performance of Sorcery, we conduct simulation studies with different network, peer, content and execution models. The evaluation results show that Sorcery can effectively detect the deceivers in system, and make the normal users avoid from downloading the corrupt or malicious content. Throughout the entire experiments, we assess the various performances of Sorcery as compared to Credence [10]—one of the most representative reputation models.

Roadmap The rest of this paper is organized as follows. The related work will be given in Section 2. The details of Sorcery are described in Section 3. We present deeper discussions of incorporating Sorcery with the schemes against MITM, Sybil, DoS and content integrity attacks in Section 4. Section 5 shows the simulation methodology and evaluation results. Finally, we give our conclusions in Section 6.

2 Related work

Along with the extensive deployment of P2P content sharing systems, security problems, especially the deceptive votes, have gradually been considered as the bottleneck of the further development of such systems.

[1] The definition of the tricky deceiver can be found in Section 5.3 or [16]

So far, many reputation models have been proposed to resist the deceptive voting behavior in P2P content sharing systems. In general, these reputation models can be grouped into three categories: *peer-based* reputation models, *object-based* reputation models and *hybrid* reputation models.

Peer-based reputation models In peer-based reputation models, e.g., EigenTrust [14], PeerTrust [13] and Scrubber [9], in order to reflect the level of honesty, each participator is assigned a reputation score by considering his past behaviors in pairwise transactions. According to the reputation score, genuine users could identify deceivers, and then isolate these deceivers from the system. However, the studies in [10, 41] evaluated the potential impact of peer-based reputation models, and implied that these models are insufficient to defend against the deceptive voting behavior.

Object-based reputation models Among the object-based reputation models, Credence [10] is the typical representative of them. In Credence, genuine users determine the reputation score of object through secure tabulation and management of endorsements from other users. To reduce the probability of believing the votes of deceivers, Credence utilizes the statistical correlation to measure the reliability of users' past votes, and designs a decentralized flow-based trust computation to discover trustworthy users. However, a newcomer without voting history hardly has any capacity of distinguishing between authentic votes and deceptive votes.

Hybrid reputation models Aiming at combining the benefits of both peer-based and object-based reputation models, several hybrid reputation models, e.g., XRep [8], X^2Rep [7] and Extended Scrubber [30], have been further presented. XRep and X^2Rep extend the work in [27] by additionally computing the reputation of object with the weights based on the past voting behavior of peers.

Besides the reputation models, several other deceptive voting behaviors defenses have been proposed in the context of P2P networks. Micropayment techniques, e.g., MojoNation [2] and PPay [26], can be utilized to counter the deceptive votes by imposing a cost on deceivers—to inject corrupt content into the system they should first commit a certain amount of resources. Furthermore, the fair exchange protocol [28] provides the mechanism similar with the micropayment to eliminate the benefit gained by deceivers in P2P networking systems.

To the best of our knowledge, none of the previous work focused on using the challenge-response-like approach to address the problem of deceptive voting behaviors in P2P content sharing systems.

3 Sorcery

In a typical P2P content sharing system, clients publish some *content*, and each content has a specific *title*. Normally, a title has various *versions*,[2] each of which is shared by a group of *providers*. Without loss of generality, this paper defines *content item* as the specific version associated with a designated title. Moreover, the peers who have voted on a specific content item are called the content item's *content voters*. Note that the content provider may have not voted on some of his shared content items; also, the content voter may have removed the content items which were voted by himself.

Generally, a client issues queries to the system in the form of keywords. The providers respond by sending the matched content items back to the client. Here, the client needs to judge the authenticity of each content item before attempting to obtain it, since some may contain malicious or corrupt data. Due to the lack of reliable evidence for reference, a client usually makes the downloading decision based on the votes on this content item.

Sorcery helps the clients judge the authenticity of votes on the *target content items*, and reduces the malicious impact brought by the deceivers. Before giving the design rationale, we first describe the infrastructure of Sorcery.

Infrastructure of Sorcery Sorcery introduces the social network to the P2P content sharing system, and thus each client may establish the friend-relationships with some users in the system. A Sorcery client needs to maintain two lists:

- *Friend list* This list contains the information of client's friends (e.g., friends' content items, votes, etc).
- *Respondent list* This list is comprised of the peers who have ever been challenged by the client. Note that client's friend list can not be seen by other peers. Specifically, as shown in Fig. 1, *Bob* can not see *Alice*'s friend list, and *Friend*1 or *Friend*2 also can not see *Alice*'s friend list. For *Friend*1 or *Friend*2, he only knows himself is one of *Alice*'s friends.

[2]The definition of the terms title and version can be found in [29].

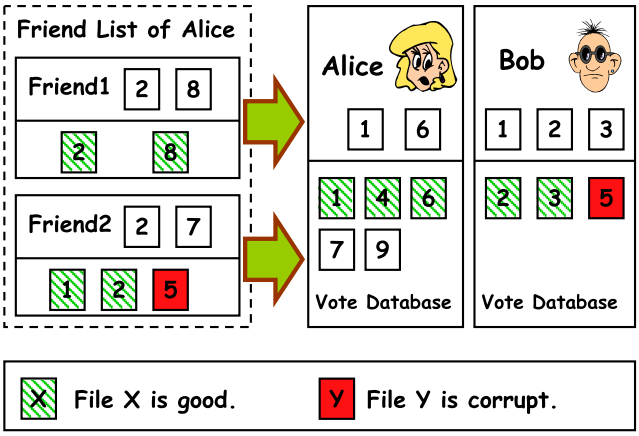

Fig. 1 Friend-relationships. To simplify the description, the *File* represents the specific content item, and we denote the content item X as *File*X

Using confidential friend information, client constructs the dominant information, which is reflected during challenge-response (Section 3.2), with respect to the other peers in the system. Moreover, each peer need to maintain a *vote database* to store the peer's own votes.

3.1 Social network

Sorcery introduces the social network to the P2P content sharing systems. In this section, we describe the design of the social network of Sorcery.

Confidential friend information In Sorcery, each client has many friends and stores their information in his own friend list. The friend list of client is confidential to other peers in the system; therefore, Sorcery constructs the dominant information for each client with the confidential friend information. As shown in Fig. 1, *Alice* publishes *File*1 and *File*6, and moreover her votes on *File*1, *File*4, *File*6, *File*7 and *File*9 are stored in her local vote database. In this instance, *Alice* has two friends in her friend list, and he can see his friends' information, e.g., content, votes, etc. But, *Bob* cannot obtain any information of *Alice*'s friends. Moreover, *Alice* will timely update the information of own friend list.

Establishment of friend-relationships Any peer can be invited by an existing participant in the network, and thus added into the system; meanwhile the inviter will become the friend of the newcomer automatically. The invitation scheme ensures that the client at least owns one friend in the system. Besides the method of invitation, the client can establish friend-relationships with the peers who are the real-world acquaintances, or the online friends recognized in other social networks. For Sorcery, the friend-relationship is symmetric, and a peer needs to send a request to the other peer for adding himself as a friend, and then the friend relationship can be established after the other side's agreement. The fundamental reason of utilizing the friends' information is that they are much more trustworthy than the anonymous peers in the system; however, the client's friends may be malicious or compromised, thus we will present a mechanism to address this practical issue, in Section 3.4.2.

Effectiveness Due to introducing social network to the P2P content sharing system, Sorcery can address the "cold start" problem. This problem means, when a newly incoming user without any voting history joins in the system, he can easily be deceived due to the lack of enough experiences of interactions with others. The studies in [16, 21] indicated that the existing reputation models cannot completely address the "cold start" problem. However, in Sorcery, once joining in the system, the newcomer first establish his friend-relationship quickly using the social network, and collect the votes from his friends. Specifically, these "shortcut" votes can be treated as the newcomer's initial local voting history; thus, we can say the newcomer quickly obtains the experiences as a mature participant in the system.

3.2 Challenge-response mechanism

In this section, we describe the details of challenge-response mechanism. Besides the friend list, each client also maintains a respondent list which stores the *reliability degrees* of the response peers who have been challenged by the client. Based on honesty of the response peer, the client computes the reliability degree with respect to the response peer (The calculation of reliability degree will be given in Section 3.3).

After the client issues a search with some keywords, the system returns with the matched content items, and ranks the search results in descending order based on the reliability degrees of the providers. If the target content item is owned by the client's friends, he can download the content item from his friends directly. However, in the most cases, the client's friends do not have the target content item. Therefore, the client should choose some of target content providers, based on their orders, to perform challenge-response. To elaborate the process of challenge-response clearly, we define $\{Vote_{j(i)}\}_{i=1}^{n}$ as the set of overlapping votes of the client's friends and the provider j, where n denotes the size of the set of overlapping votes, and $Vote_{j(i)}$ denotes the vote of the provider j on the content item i. Then,

we define $\{C_i\}_{i=1}^n$ as the set of content items associated with the elements of $\{Vote_{j(i)}\}_{i=1}^n$. The element of $\{C_i\}_{i=1}^n$, C_i, denotes the content item associated with the $Vote_{j(i)}$. The process of challenge-response is as follows:

- *Step 1—Challenge* The client first generates the *challenge message* which is comprised of the queries for some content items. Here, the query denotes the voting request on a specific content item C_i. Meanwhile, the client inserts the query for the target content item into the challenge message randomly for confusing the target content provider. Note that the reason that we do not use the queries for all the elements of $\{C_i\}_{i=1}^n$ is to avoid the target provider from knowing the details of $\{Vote_{j(i)}\}_{i=1}^n$ and judging the client's target content item next time. Afterwards, the challenge message is sent to the target content provider. In a similar way, the client also generates the challenge message to challenge other providers of the target content item.
- *Step 2—Response* After receiving the challenge message, the provider should respond the client with the *response message* which is comprised of local votes for the queries in the challenge message. These votes contain the provider j's votes on each C_i and the target content item. Similarly, other providers also need to respond the challenges in the same way.

When the client receives the responses, he may estimate whether to believe each response peer's vote on the target content item based on ϑ—the rate of each response peer's correct answers. This rate can be set according to the different requirements of applications. Normally, we think $\vartheta \leq 0.5$ indicates weak or no correctness. Due to the feature of utilizing the correctness of response, in practical applications, Sorcery may be threaten by the target deceivers who falsely vote on the popular content items and vote correctly on other normal content items. In fact, the above threat will not harm Sorcery. Because Sorcery client first examines whether his friends own or have voted on the popular content item (target content item), we believe that, normally, the client's friends may have downloaded or voted on the popular content item. Therefore, the client could obtain directly the popular content item or the associated votes from his friends before challenging the providers of target content item.

Example To elaborate the challenge-response mechanism more clearly, we describe the whole process with the instance in Fig. 2. In this instance, *Alice* issues a search for *File*3. Because *Alice*'s friends do not have *File*3, *Alice* performs the challenge-response to some providers of *File*3 according to their orders. As shown in Fig. 2, we assume that *Bob* and *Eve* are the top2 providers of *File*3 in *Alice*'s ranking result.

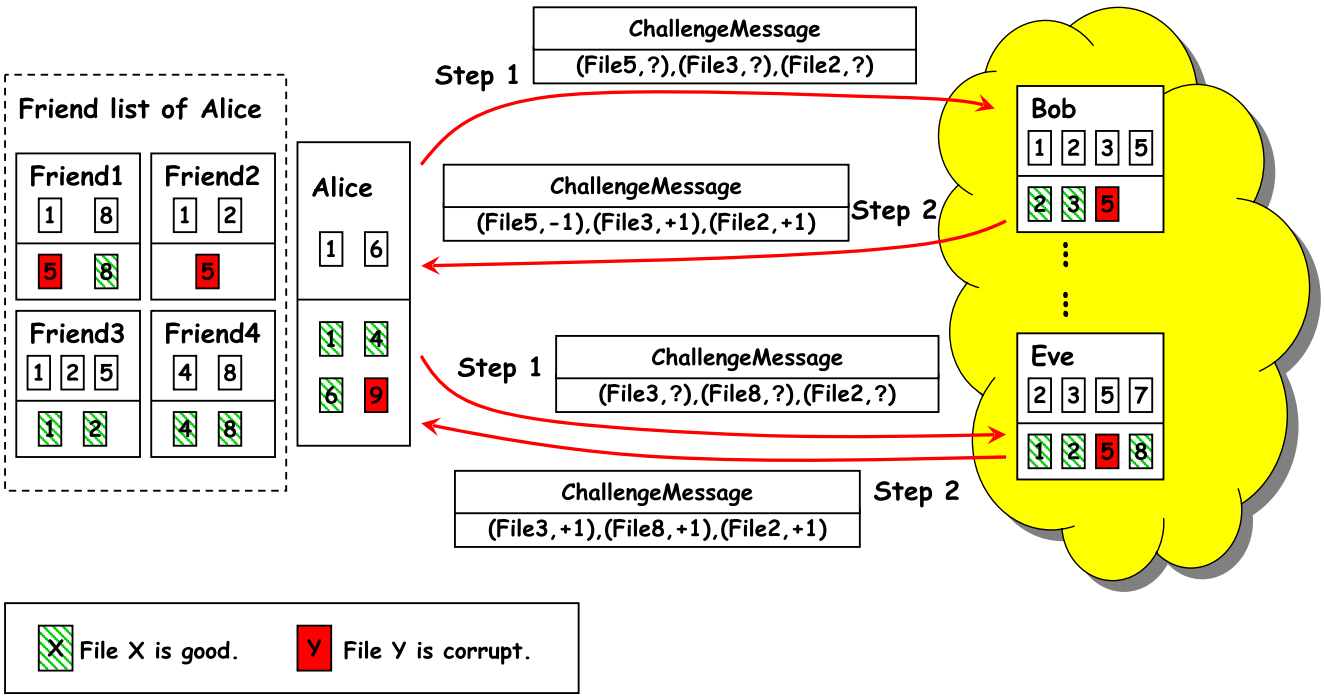

Fig. 2 Challenge-response mechanism. To simplify the description of instance, the *File* represents the specific content item, and we denote the content item X as *File*X

Therefore, *Alice* performs the challenge-response to them as follows:

- *Step 1—Challenge* *Alice* chooses the queries for *File*2 and *File*5 to generate the challenge message, since *Alice*'s friends have the votes on the two content items. Then, Sorcery client inserts the query for *File*3 into the challenge message randomly, and sends to *Bob*; likewise, *Alice* also generates the challenge message to challenge *Eve*.
- *Step 2—Response* After receiving the challenge message, *Bob* should "answer" the *Alice*'s challenge with his own votes on *File*2, *File*3 and *File*5, and then returns the response message to *Alice* as Fig. 2 shown. Similarly, *Eve* also needs to respond the challenge of *Alice* in the same way.

After *Alice* receives the responses, the challenge-response mechanism will tell *Alice* whether to believe the votes of *Bob* and *Eve* on *File*3. However, in some practical cases, there are possibly two issues as follows:

1. How to handle the peers challenged by the client do not provide responses?
2. How to handle the situation that there are no overlapping voting histories of both client's friends and target content providers?

Answering the first issue For the peers challenged by the client, if they do not provide responses in a certain time internal,[3] clients will choose to treat them as the deceivers. Although some users cannot provide responses due to network problems, we find the above scheme is reasonable according to our experimental results. The experimental results shown in Section 5 prove this solution provides an interested incentive mechanism for Sorcery clients.

Answering the second issue For the second problem, we utilize the measurement results of real-world P2P content sharing system to design solution. We will discuss the details of this solution in Section 3.4.1.

3.3 Punishment mechanism for deceivers

In order to reduce the possibility of impact brought by deceivers, Sorcery proposes a severe punishment mechanism to the client for computing the reliability degrees of response peers. Using the punishment mechanism, the client i computes the reliability degree, $RD_{i(j)}$, with respect to the response peer j according to each judgemental result for the peer j as follows:

$$RD_{i(j)} = \begin{cases} \max(-1,\ RD_{i(j)} - pn^2) & \text{if } j \text{ is a deceiver} \\ \min(1,\ RD_{i(j)} + r) & \text{otherwise} \end{cases} \tag{1}$$

where

- n The number of peer j's response being judged as deceptive behavior.
- p The penalty factor given to the peer j.
- r The recompense factor given to the peer j.

In Eq. 1, if a deceptive peer is detected, his reliability degree will be decreased quickly. Sorcery allows the peers recover their reliability degrees by responding genuinely. We propose to set $p > r$, and thus the reliability degree can decrease faster than it increases. For a strange response peer, his initial reliability degree is set to 0.

Because each search result is ranked according to the content providers' reliability degrees, the content items provided by the deceivers will be placed at the end of search result. Therefore, by degrading the rank of the response content items, Sorcery reduces the impact brought by the deceivers. Besides the demotion-based punishment, when receiving a search request from the peer who has the negative reliability degree, the client should ignore the request.

To spread the impact of punishment, after computing the reliability degree, the client will broadcast the new reliability degree to his friends and friends-of-friends; meanwhile, the reliability degrees computed by both the client's friends and friends-of-friends can also be received by the client. Therefore, the power of punishment should be expanded in the reliable "social group". The study in [12] demonstrated both the reliability and security of the friends-of-friends in the real-world social networks. Therefore, we can believe the reliability degrees provided by the patulous friends.

3.4 Practical issues

This section mainly discusses how to address three important practical issues:

- The client's friends do not have the overlapping voting histories with the content providers.
- The friends of client are unreliable or compromised.
- The malicious user who correctly responds the challenge, but to transfer with bogus content item.

[3] The time internal should be set based on the concrete requirement of application.

3.4.1 Lack of the overlapping votes

Due to the lack of common interests with the client's friends, the content providers possibly do not have the overlapping votes with the client's friends. The studies in [6, 11] indicated that it is a high proportion that most peers in the system have overlapping votes with the voters of any content item. Therefore, when the client's friends do not have the overlapping votes with the content providers, Sorcery client seeks for the target content voters, and ranks these voters. The ranking score of the client i with respect to the voter j, $RS_{i(j)}$, is computed as follows:

$$RS_{i(j)} = VN_j \times RD_{i(j)} \tag{2}$$

where

VN_j The total number of the votes of voter j.
$RD_{i(j)}$ The reliability degree of the client i with respect to the voter j.

According to the order of voters, the client will challenge some of these voters. As shown in Fig. 3, we assume that *Carol* and *Dave* are the top2 voters of *File*3 in the *Alice*'s ranking result, and they do not have the *File*3. According to the result of challenging *Carol* and *Dave*, *Alice* can judge that *Carol* is a genuine user and *Dave* is a deceiver. Thus, *Alice* may believe the *Carol*'s vote on *File*3, and download *File*3 from the system; otherwise, *Alice* will challenge other voters based on the order of voters.

Discussion Because the distribution of content in the actual P2P content sharing systems follows Zipf with the parameter $\alpha = 0.8$ [6], the previous measurement studies in KaZaA [3] reported an interesting fact that it always be much high probability that the most peers in system should always have the overlapping votes with those voters of popular content [6, 19]. Even for the voters of normal content items, the studies in [10, 11] also provided the real-world evidence that any peer in system still has the overlapping votes with these voters with high probability. Therefore, we believe that the client's friends have the overlapping votes with the target voters, with relatively high probability. In other words, we can always utilize the overlapping votes between client's friends and target content voters to generate challenge message.

However, it indeed exists the instance that the client's friends have no overlapping votes with all the voters of target content. To address such problem, Sorcery client should leverage the transitivity of social network, e.g., the friends-of-friends, to amplify the voting set which should be used to generate the challenge.

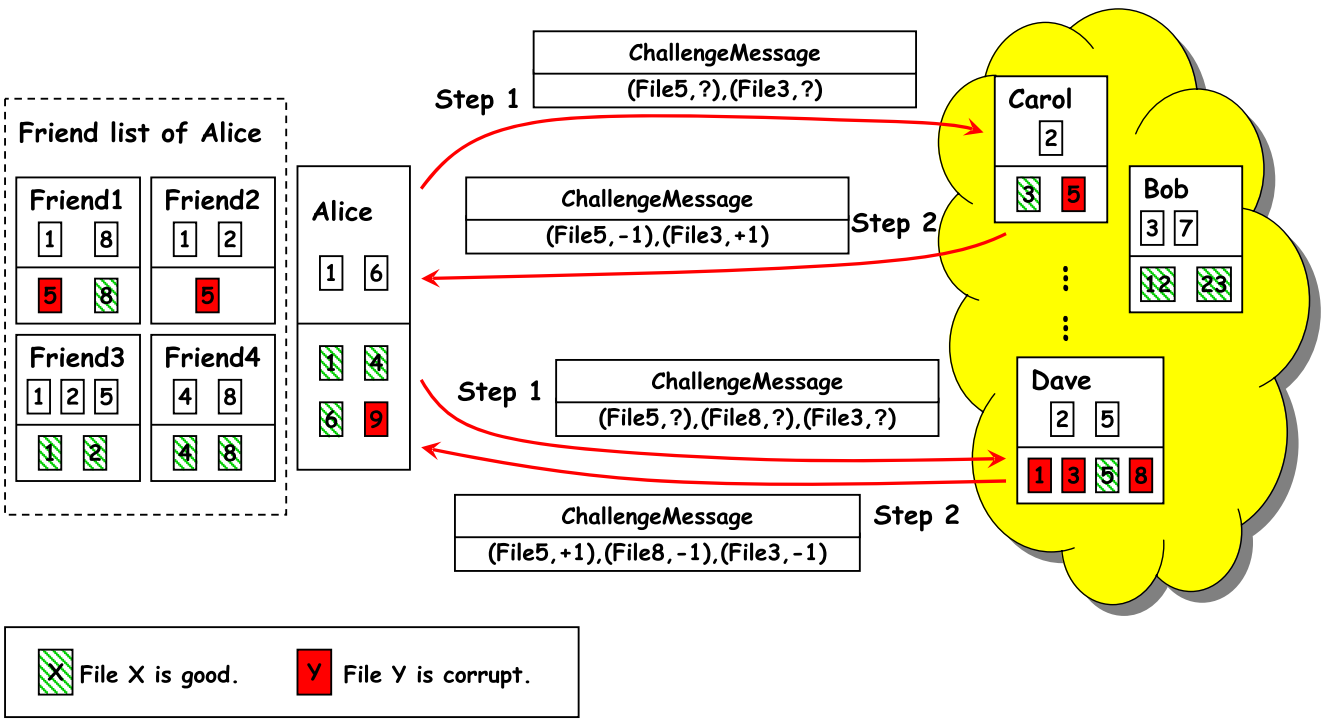

Fig. 3 Challenge-response to the content voters. To simplify the description of instance, the *File* represents the specific content item, and we denote the content item X as $FileX$

Meanwhile, Sorcery can also utilizes the patulous reliability degrees (mentioned in Section 3.3) to help the client judge the authenticity of target content. Note that because the study in [12] has demonstrated both the reliability and security for the friends-of-friends, we believe that the use of transitive social network is reasonable. In addition, to bound the harms incurred by the Sybil attacks during leveraging the transitivity of social network, we could adopt SybilLimit [35] approach based on the common insight on social network with our work. As an alternative, SumUp [16] is also a good scheme. In our experiments, we also demonstrate that the expansion of social network indeed makes Sorcery more robust to the deceivers.

3.4.2 Unreliable friends

In the practical applications, some friends may be online deceivers or compromised. Therefore, Sorcery proposes the similarity between the client i and his friend f, $Sim_{i(f)}$, to resist the harms incurred by unreliable friends. The calculation of the similarity is based on the cosine technique as follows:

$$Sim_{i(f)} = \frac{\sum_{k \in C} \left(V_{i(k)} V_{f(k)}\right)}{\sqrt{\sum_{k \in C} \left(V_{i(k)}\right)^2} \sqrt{\sum_{j \in C} \left(V_{f(j)}\right)^2}} \tag{3}$$

where

- $V_{x(y)}$ The vote from peer x to the content item y, and normally the value is $+1$ or -1. If x has not voted on the content item y, the value is 0.
- C The content set of the system.

For two users who have not casted any vote on the same content items, the similarity between them is set to 0. The client will compute the similarities of all his friends intermittently. Once the friend has the similarity lower than 0.5, the client should not utilize the votes of the friend to generate the challenge message, since the study in [11] indicated that the similarity lower than 0.5 represents weak or no correlation between two users.

3.4.3 Incredible interaction

Another serious security vulnerability is that, in the process of challenge-response, a malicious content provider may correctly respond the client's challenge, but replies with the bogus content item. This threat has been mentioned in [13] called *incredible interaction*, and Sorcery proposes the similar approach with [13], but based on the social network, to bound the impact of incredible interaction.

When the client i wants to download the content item from the provider j, he first examines his own evaluation history of interaction. If i has interacted with j, he may make decision based on his past experiences; otherwise, the client i should compute the *credibility* with respect to j, $Cred_{i(j)}$, as follows:

$$Cred_{i(j)} = \frac{\sum_{f \in F} \left(Sim_{i(f)} Eval_{f(j)}\right)}{|F|} \tag{4}$$

where

- F The set comprised of i's friends and his friends-of-friends who have evaluated the interaction with the peer j.
- $Eval_{f(j)}$ The evaluation of the interaction from f to j, and the value is $+1$ or -1, where $+1$ denotes a satisfied interaction and -1 denotes the bogus one.
- $Sim_{i(f)}$ The similarity between the client i and f.

The credibility is interpreted as an estimate of the authenticity of the content provider. This estimate should be used to make a decision to accept or reject downloading. Due to using the similarity as the weight to compute the credibility, the evaluations of the malicious friends cannot make harm. After downloading, the client needs to evaluate this interaction. These evaluations are shared to his friends and the friends-of-friends; meanwhile, the client may also obtain the evaluations shared by his friends and friends-of-friends.

Discussion The threat of incredible interaction is difficult to avoid by most existing schemes. To the best of our knowledge, most reputation models, such as EigenTrust [14] and Credence [10], cannot address this problem. On the other hand, Sorcery utilizes reliable social network to bound this threat, and adding integrity verification into the overlay network will be discussed in the following section.

4 Deeper analysis and discussion

In real-world applications, some other types of attacks can also be mounted against Sorcery, such as MITM attack, Sybil attack [22], DoS attack and content integrity

attack. In this section, we discuss how to resist these challenges.

4.1 Resisting MITM attack

Generally, an MITM attacker could read, insert and modify the messages between two sides of the challenge-response without letting any of them has the knowledge of compromised transaction between them. Therefore, both the challenge and response generated by the two sides may be unauthenticated. To resist such MITM attack, the peers should dynamically maintain a trusted group to perform multiple independent exchanges originating from the different trusted group members (similar to the mechanism described in [33]); then, the peer can execute the Byzantine agreement protocol [38] to obtain the actual messages.

4.2 Resisting Sybil attack

Another important security vulnerability is the Sybil attack—a deceiver takes on multiple identities and pretends to be distinct peers [22]. Under Sybil attacks, the challenge-response mechanism is easy to be compromised. Based on the same insight, Sorcery clients could directly utilize their social networks to limit the deceptive voting behaviors of Sybil attackers as the approaches in studies [34, 35]. As an alternative, we could also adopt the computational puzzle scheme against the Sybil attack [36, 37].

4.3 Resisting DoS attack

In general, DoS attack is the serious threat where one or more malicious users attempt to thwart genuine users from having access to legitimate services. In our work, the popular content providers may be flooded by huge amounts of challenge, i.e., Sorcery may suffer from the DoS attack. The study in [32] indicated that the ingenious solution against DoS attack is based on the calculation of puzzle scheme. Therefore, to resist DoS attack, each Sorcery client may compute moderate expense, but not intractable puzzles to gain the admission to challenge those popular content providers. Furthermore, based on the similar fact that it is difficult for a user to create arbitrarily many trust links, we can utilize the Ostra system [45] explored the use of existing social links to impose a cost on DoS attackers, thus preventing the adversary from sending excessive unwanted communication. Recently, some interested mechanisms, e.g., Not-a-Bot [43] and BotGraph [42], can also be used to incorporate Sorcery against DoS attack.

4.4 Protecting the integrity of content items

We have mentioned that the incredible interaction is a tough problem for the existing security mechanisms in P2P content sharing systems. If we incorporate Sorcery with some integrity verification schemes, we can make Sorcery more robust to the threat of incredible interaction. Sorcery can make use of the framework proposed by Habib et al. [44] to verify content integrity, and this framework could provide high assurance of data integrity with low computation and communication overheads. As an alternative, we could also utilize the security framework introduced by Chen et al. [31] to accurately distinguish polluted content items and verify the integrity of the requested content items. Because the overheads of the above two integrity verification frameworks are both low, we believe they can make Sorcery more robust to the problem of incredible interaction.

5 Evaluation

This section is organized as follows:

- First, we describe the simulation setup including network, peer, content and execution models;
- Then, we present the key performance metric of our evaluation;
- Finally, we evaluate Sorcery as compared with the Credence [10] with various experimental configurations.

5.1 Simulation setup

To evaluate the performance of Sorcery, we developed a P2P content sharing prototype system with all the mechanisms of Sorcery. Moreover, we generate several models with different parameters—follow the certain distributions. Table 1 shows the important parameters throughout our simulations.

Network model In the following simulations, we choose Gnutella [1] as the underlying overlay network of Sorcery. Because our focus is on the dissemination of the content in the network, we assume the perfect overlay routing and content discovery. Furthermore, the transfer time is assumed to be negligible.

Peer model The network is composed of 5,000 peers, and there are two categories of peers, genuine peers and deceivers, in our simulation. At the startup, the genuine peers only publish the good content items, and correctly vote on the content items; whereas, the

Table 1 Experimental configurations

Parameter	Meaning
PC	The correctness rate that the genuine peer votes on the content items.
PD	The probability that the deceiver votes on the content items correctly.
PR	The probability that the peer gives response when he is challenged.
PV	The probability that the peer gives votes on the content items.
FC	The total proportion of collusive deceivers.
FD	The total proportion of the deceivers.
FN	The number of each client's friends.

deceivers share the corrupt content items, and give the positive votes on them. Throughout our simulation, two categories of peers may download content items, leave and rejoin the system. We generate the social network of simulation according to small world property of online social networks [40], and establish the friend-relationships for the peers based on the widely adopted Kleinberg model [39].

Content model The study in [6] indicated the existence of a large number of corrupt versions for the single file (title) in the actual system. In our simulations, there are 1,000 unique files (titles), and each of which has 500 different versions containing 50 good versions. At startup, each genuine peer publishes 30 content items and each deceiver shares 200 content items. Furthermore, the versions published by a peer are determined by first selecting a certain title and then its version. Specifically, both selections follow Zipf distribution with the parameter $\alpha = 0.8$ [6].

Execution model Different queries are initiated at uniformly distributed peers in the overlay network. An experimental simulation is composed of 50 simulation cycles. In each cycle, the selection of 0–5 specific content items to download is done by first selecting a title and then choosing a version based on the mechanisms of Sorcery. After each simulation cycle, the number of corrupted downloads is calculated. The genuine peers and deceivers may download good and corrupt content items; however, the deceivers should give the positive vote on a corrupt content item and a negative vote on the good content item. On the other hand, a genuine peer should give the votes, with the correctness PC, on the content items in the system. Without the especially emphasized, we set $PC = 0.9$, $PD = 0$, $PR = 1$, $PV = 1$, $FC = 0$, $FD = 0.2$, and $FN = 6$ as the default configurations of our simulations. Each experimental simulation is run 5 times and the results of all runs are averaged.

5.2 Performance metric

In the following simulations, we characterize the system performance with the *rate of good downloads*. It is defined as the rate of downloads that the clients acquire good content items in one simulation cycle. Specifically, this metric is computed at the end of each simulation cycle.

5.3 Experimental results

Recently, most of the defense mechanisms against deceptive voting behaviors deployed in P2P content sharing systems are based on the reputation models. Among these models, Credence [10] is an ingenious model deployed on a real-world network; moreover, its scenario is similar to Sorcery. Therefore, we compare the performance of Sorcery with that of Credence throughout our experiments.

Impact of the probability of peers voting Due to the feature that both Sorcery and Credence need the votes from the peers, we compare Sorcery with Credence under the different PVs. As shown in Fig. 4, when the PV is set to 1.0, the rate of good downloads of both Sorcery and Credence can increase to higher than 0.7 after 5 and 10 cycles respectively. However, when setting PV to 0.2, we notice that both the performances of Sorcery and Credence are affected by this low probability of peer voting—the rates of good downloads of two models are always lower than 0.7 during 50 simulation cycles. Therefore, we conclude that these two models strongly depend on the cooperation of peer voting, and Sorcery can work with the better *convergence* due to the reliable votes provided by the clients' friends. Here,

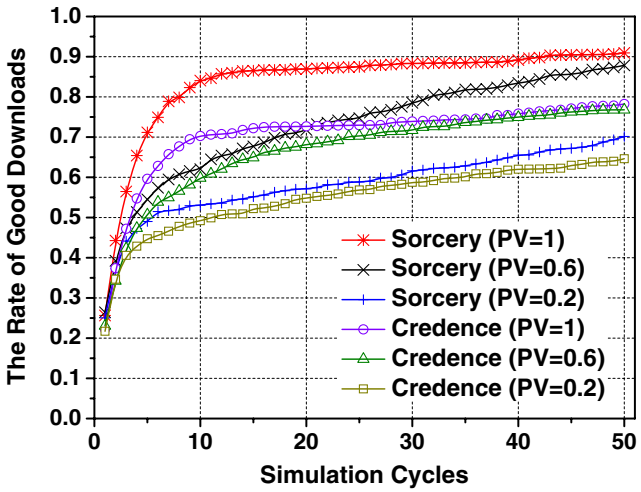

Fig. 4 Impact of the probability of peers voting

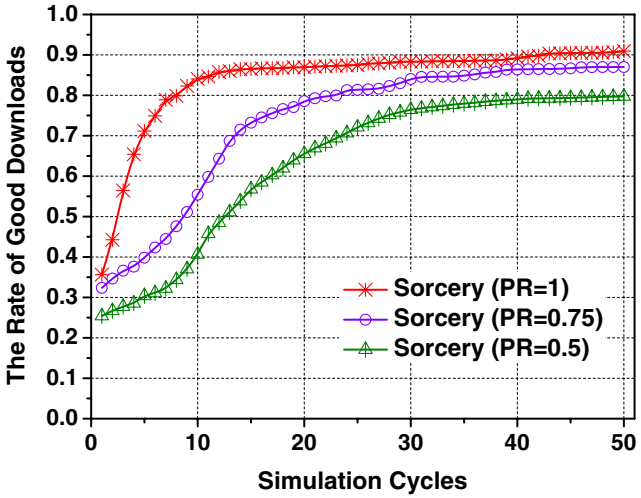

Fig. 5 Impact of the probability of peers responding

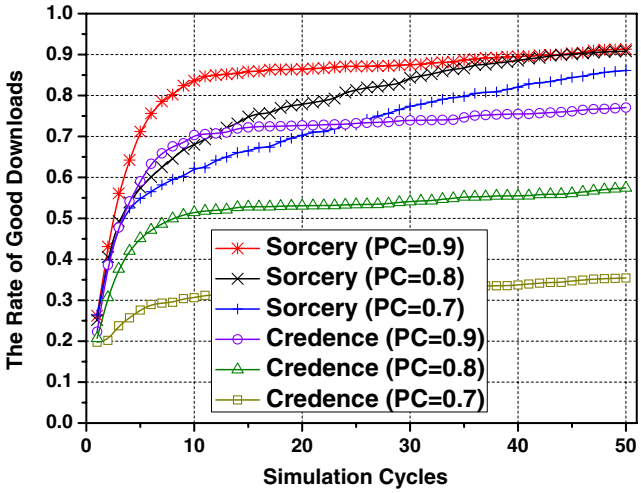

Fig. 6 Impact of the correctness of voting

convergence denotes how long the system takes to reach its maximum performance (The rate of download to good content).

Impact of the probability of peers responding Figure 5 shows the rate of good downloads of Sorcery under the conditions that PR is set to 1.0, 0.75 and 0.5 respectively. It is clear that, when $PR = 1.0$, the rate of good downloads can quickly converge to 0.8 in only 8 simulation cycles. Even if PR is decreased to only 0.5, Sorcery can still turn the rate of good downloads to above 0.7 after 20 simulation cycles. This experimental result indicates that, in order to construct the good download rate of content in the system, the peers should actively respond each other. Meanwhile, this result reflects the incentive mechanism of Sorcery—the more actively users respond, the better performance they can obtain.

Impact of the correctness of voting In this simulation, we evaluate the performance of Sorcery compared with Credence under different correctness rates of the genuine peers' votes (PC). This experiment is based on the consideration that, in the real-world applications, some genuine users cast several incorrect votes on the content in system by mistake. As the results shown in Fig. 6, it is clear that, the change of PC makes a strong influence on both the performances of Sorcery and Credence. We notice that Sorcery can always work better than Credence under three different values of PCs. Therefore, we conclude that Credence is more vulnerable to PC than Sorcery.

Impact of the normal deceivers The simulation in Fig. 7 simulates the impact of total proportion of deceivers (FD) on the performances of the simulated two models. Throughout the simulations, we assume the system will not be attacked by P2P worms, so that, the percentage of deceivers should generally not be higher than 30%. Interestingly, the result in Fig. 7 shows that the different FDs do not significantly influence the performances of Sorcery and Credence. The reason is that, although the proportion of deceivers is increased, Sorcery client still utilizes the votes of his own friends to challenge other peers respectively. Therefore, the performance of Sorcery cannot be affected by the proportion of deceivers. For Credence, because peers' downloads are based on their own judgements of the content authenticity, Credence clients cannot be affected along with the increase of deceivers.

Impact of the tricky deceivers In Fig. 8, we evaluate the performances of Sorcery and Credence against the

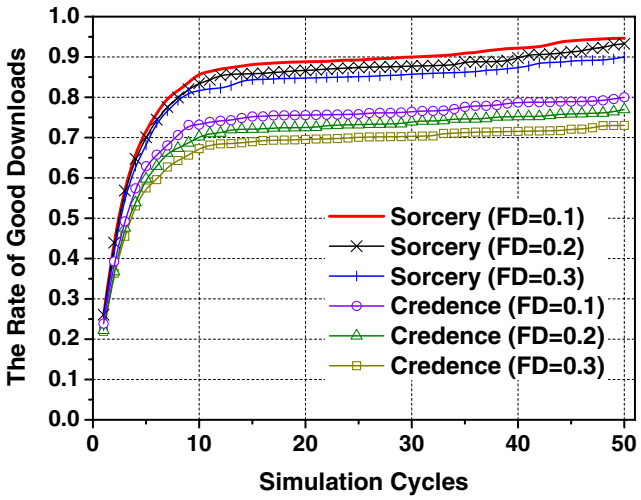

Fig. 7 Impact of the normal deceivers

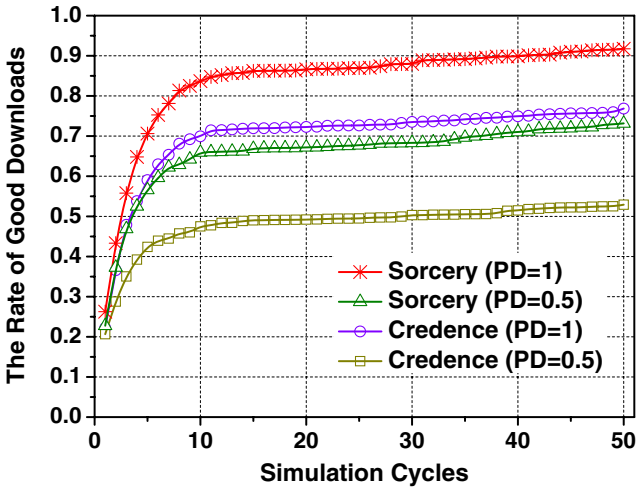

Fig. 8 Impact of the tricky deceivers

tricky deceivers. Tricky deceivers are the malicious users who can pretend the genuine users by casting correct votes on some content items, thus deceiving the normal users. Using this type of attack, the tricky deceivers may attack some specific content items by giving opposite votes. Figure 8 indicates that, when confronted with tricky deceptive voting behavior, Credence under $PD = 0.5$ works a worse performance, which is approximately reduced by 40%, than the Credence under the condition without tricky deceivers ($PD = 1$). Sorcery is also influenced by tricky deceivers; however, as shown in Fig. 8, although the performance of Sorcery is influenced by tricky deceivers when setting PD to 0.5, Sorcery can work better than Credence.

Impact of the collusive deceivers The results shown in Fig. 9 demonstrate, under the collusive attacks, Sorcery can work much better than Credence. The reason is that, for Sorcery, each client generates *challenge messages* using the overlapping votes of both his friends and target voters; therefore, for different target voters, the challenge messages generated by the client are different. Moreover, because challenge-response happens between the client and individual target voter, the collective deceptive voting behaviors of collusive deceivers cannot affect the performances of Sorcery clients badly. From the above two experiments (Impacts of Tricky and Collusive Deceivers), we deduce that the key reasons that makes Sorcery outperform Credence are active challenge-response mechanism and the utilization of social network. This result clearly demonstrates the conclusion in the study [15]—the social network can make the P2P sharing systems more robust.

Impact of peer's friend number The above simulations have reflected the impact of social network. In this simulation, we discuss the impact of each peer's friend number (FN). Figure 10 shows an interested phenomena. When setting FN to 2, 4, 6 and 8 respectively, the performance of Sorcery changes a lot. When each peer only has two friends ($FN = 2$), the rate of good downloads is always lower than 0.4 during 50 simulation cycles; however, as setting FN to 6, Sorcery can perform robustly to the good downloads. Interestingly, when we vary the FN to 8, the enhancement of performance is not so prominent as the previous experiments (FN is set to 2, 4 and 6). This result indicates that a Sorcery client actually does not need to hold a large number of friends.

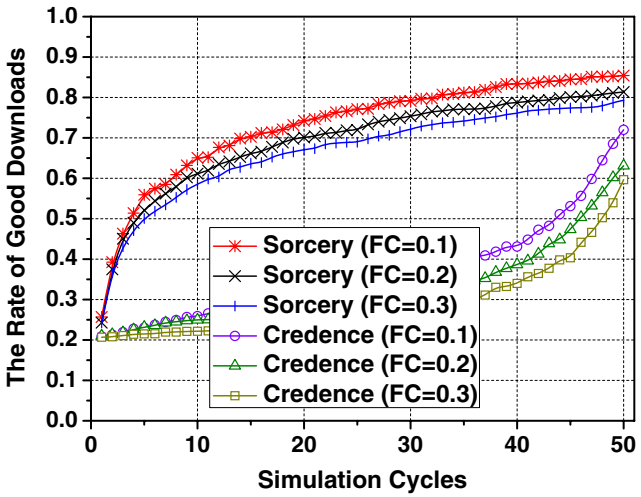

Fig. 9 Impact of the collusive deceivers

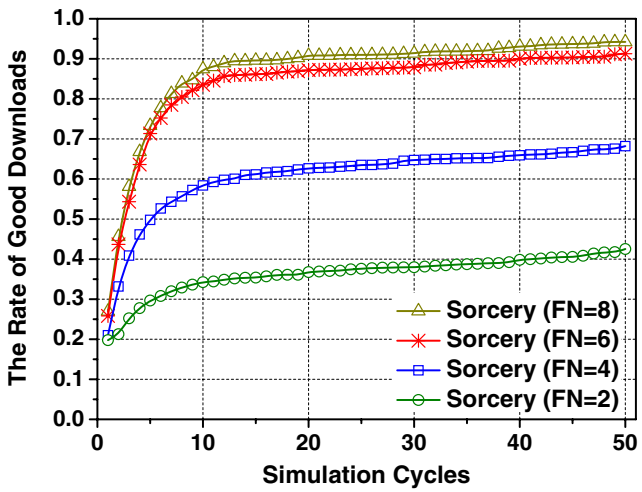

Fig. 10 Impact of peer's friend number

6 Conclusion

This paper presents Sorcery, a novel challenge-response approach against deceptive voting behaviors in P2P content sharing systems. Sorcery relies on three key mechanisms: (1) introducing social network into current P2P content sharing system; (2) utilizing the overlapping voting histories of both client's friends and the content provider/voter to challenge the latter actively, thus judging whether the other side is a deceiver; (3) using punishment mechanism to reduce the impact brought by deceivers. Our experimental results illustrate that Sorcery can effectively address the problem of deceptive voting behaviors in the P2P content sharing systems, and work better than the existing reputation models.

References

1. Gnutella. Available at: http://www.guntellaforums.com
2. MojoNation. Available at: http://sourceforge.net/projects/mojonation/
3. KaZaA. Available at: http://www.kazaa.com
4. eMule. Available at: http://www.emule-project.net
5. BitTorrent. Available at: http://www.bittorrent.com
6. Liang J, Kumar R, Xi Y, Ross KW (2005) Pollution in P2P file sharing systems. In: Proceedings of IEEE INFOCOM. Miami, FL, USA, March 2005
7. Curtis N, Safavi-Naini R, Susilo W (2004) X^2Rep: enhanced trust semantics for the Xrep protocol. In: Proceedings of ACNS, pp 205-219
8. Damiani E, De Capitani di Vimercati S, Paraboschi S, Samarati P, Violante F (2002) A reputation-based approach for choosing reliable resources in Peer-To-Peer networks. In: Proceedings of ACM conference on computer and communications security (CCS'02), pp 207–216
9. Costa CP, Soares V, Almeida JM, Almeida V (2007) Fighting pollution dissemination in Peer-To-Peer networks. In: Proceedings of ACM SAC, Seoul, Korea, pp 1586–1590
10. Walsh K, Gün Sirer E (2006) Experience with an object reputation system for Peer-to-Peer filesharing. In: Proceedings of NSDI, May 2006, pp 1–1
11. Walsh K, Gün Sirer E (2005) Fighting Peer-to-Peer SPAM and decoys with object reputation. In: Proceedings of workshop of the economics of Peer-to-Peer systems, August 2005, pp 138–143
12. Garriss S, Kaminsky M, Freedman MJ, Karp B, Mazieres D, Yu H (2006) RE: reliable email. In: Proceedings of NSDI San Jose, California, USA, 8–10 May 2006
13. Xiong L, Liu L (2004) PeerTrust: supporting reputation-based trust for Peer-to-Peer electronic communities. In: IEEE transaction on knowledge and data engineering Knowledge and Data Engineering, IEEE Transactions on, vol. 16, no. 7, pp 843–857
14. Kamvar SD, Schlosser MT, Garcia-Molina H (2003) The eigentrust algorithm for reputation management in P2P networks. In: Proceedings of WWW Budapest, Hungary, pp 640–651
15. Pouwelse J, Garbacki P, Wang J, Bakker A, Yang J, Iosup A, Epema D, Reinders M, van Steen M, Sips H (2006) Tribler: a social-based Peer-to-Peer system. In: Proceedings of IPTPS Santa Barbara, CA. February 2006
16. Tran N, Min B, Li J, Subramanian L (2009) Sybil-resilient online content voting. In: Proceedings of NSDI, pp 15–28
17. Fudenberg D, Tirole J (1991) Game theory. MIT Press, Cambridge, MA
18. Osborne M, Rubinstein A (1994) A course in game theory. MIT Press, Cambridge, MA
19. Saroiu S, Gummadi P, Gribble S (2002) A measurement study of Peer-to-Peer file sharing systems. In: Proceedings of MMCN, San Jose, CA
20. Gummadi K, Dunn R, Saroiu S, Gribble S, Levy H, Zahorjan J (2003) Measurement, modeling, and analysis of a Peer-to-Peer file-sharing workload. In: Proceedings of ACM SOSP. In SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles, pp 314–329
21. Resnick P, Kuwabara K, Zeckhauser R, Friedman E (2000) Reputation systems. In: Proceedings of communications of the ACM 43(12):45–48
22. Douceur JR (2002) The sybil attack. In: Proceedings of IPTPS, Cambridge, MA. pp 251–260
23. Zhang H, Goel A, Govindan R, Mason K, Roy BV (2004) Making eigenvector-based reputation systems robust to collusion. In: Proceedings of workshop on algorithms and models for the web-graph. pp 92–104
24. Thommes R, Coates M (2005) Epidemiological models of Peer-to-Peer viruses and pollution. In: Proceedings of technical report. McGill University
25. Vishnumurthy V, Chandrakumar S, Sirer EG (2003) KARMA: a secure economic framework for P2P resource sharing. In: Proceedings of workshop on the economics of Peer-to-Peer systems (IPTPS'03), Berkeley, CA
26. Yang B, Garcia-Molina H (2003) PPay: micropayments for Peer-to-Peer systems. In: Proceedings of ACM conference on computers and communications security (CCS'03), 27–30 October, 2003, Washington DC
27. Cornelli F, Damiani E, di Vimercati SDC, Paraboschi S, Samarati P (2002) Choosing reputable servents in a P2P network. In: Proceedings of WWW, Honolulu, Hawaii, USA, pp 376–386
28. Gauthier P, Bershad B, Gribble SD (2004) Dealing with cheaters in anonymous Peer-to-Peer networks. In: Proceedings of technical report of University of Washington
29. Liang J, Naoumov N, Ross KW (2005) Efficient blacklisting and pollution-level estimation in P2P file-sharing systems. In: Proceedings of AINTEC
30. Costa CP, Almeida JM (2007) Reputation systems for fighting pollution in peer-to-peer file sharing systems. In: Proceedings of peer-to-peer computing (P2P'07), pp 53–60
31. Chen R, Lua EK, Crowcroft J, Guo W, Tang L, Chen Z (2008) Securing peer-to-peer content sharing service from poisoning attacks. In: Proceedings of peer-to-peer computing (P2P'08), pp 22–29
32. Parno B, Wendlandt D, Shi E, Perrig A, Maggs BM, Hu Y-C (2007) Portcullis: protecting connection setup from denial-of-capability attacks. In: Proceedings of SIGCOMM, August 2007
33. Chen R, Guo W, Tang L, Hu J, Chen Z (2008) Scalable byzantine fault tolerant public key authentication for Peer-to-Peer networks. In: Proceedings of Euro-Par, Las Palmas de Gran Canaria, Spain, pp 601–610
34. Yu H, Kaminsky M, Gibbons PB, Flaxman A (2006) Sybilguard: defending against sybil attacks via social networks. In: Proceedings of SIGCOMM, Pisa, Italy, September 2006
35. Yu H, Gibbons PB, Kaminsky M, Xiao F (2008) SybilLimit: A near-optimal social network defense against sybil attacks.

In: Proceedings of IEEE symposium on security and privacy (S&P'08), pp 3–17
36. Borisov N (2006) Computational puzzles as sybil defenses. In: Proceedings of peer-to-peer computing (P2P'06), pp 171–176
37. Rowaihy H, Enck W, McDaniel P, La Porta T (2007) Limiting sybil attacks in structured P2P networks. In: Proceedings of INFOCOM, 6–12 May 2007
38. Lamport L, Shostak RE, Pease MC (1982) The Byzantine generals problem. In: ACM transactions on programming languages and systems, pp 382–401
39. Kleinberg JM (2000) The small-world phenomenon: an algorithm perspective. In: Proceedings of STOC, pp 163–170
40. Mislove A, Marcon M, Gummadi PK, Druschel P, Bhattacharjee B (2007) Measurement and analysis of online social networks. In: Proceedings of internet measurement conference (IMC'07), San Dirgo, CA
41. Dumitriu D, Knightly EW, Kuzmanovic A, Stoica I, Zwaenepoel W (2005) Denial-of-service resilience in Peer-to-Peer file sharing systems. In: Proceedings of SIGMETRICS, pp 38–49
42. Zhao Y, Xie Y, Yu F, Ke Q, Yu Y, Chen Y, Gillum E (2009) BotGraph: large scale spamming botnet detection. In: Proceedings of NSDI, pp 321–334
43. Gummadi R, Balakrishnan H, Maniatis P, Ratnasamy S (2009) Not-a-Bot: improving service availability in the face of botnet attacks. In: Proceedings of NSDI. Boston, MA
44. Habib A, Xu D, Atallah M, Bhargava B, Chuang J (2005) Verifying data integrity in peer-to-peer media streaming. In: Proceedings of MMCN, pp 1–12
45. Mislove A, Post A, Druschel P, Gummadi PK (2008) Ostra: leveraging trust to thwart unwanted communication. In: Proceedings of NSDI. San Francisco, CA

Huiping Sun is an assistant professor in the School of Software and Microelectronics, Peking University, China. His research interests mainly lie in identity and trust management, RFID security and privacy, Social network and P2P Security.

Sihan Qing currently a professor of Peking University and the director of information security department of the School of Software and Microelectronics, Peking University. His research interests include cryptology, network and information security, Trusted Computing and security computer systems.

Ennan Zhai received the B.E. degree from Northeastern University, China, in 2007. He is currently a master student in the School of Software and Microelectronics, Peking University, China. His research interests mainly lie in distributed systems, social network-based systems and security operating systems. More information about his research is available at http://infosec.pku.edu.cn/~zhaien/.

Zhong Chen received the Ph.D degree from Peking University, China. He is currently a professor of Peking University, the dean of the School of Software and Microelectronics at Peking University and the director of Key Laboratory of Network and Software Security Assurance at Peking University. His research interests include network and information security as well as software engineer.